UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/990,527	11/21/2001	Walter R. Smith	MFCP.88141	3082

45809 7590 11/03/2006

SHOOK, HARDY & BACON L.L.P.
(c/o MICROSOFT CORPORATION)
INTELLECTUAL PROPERTY DEPARTMENT
2555 GRAND BOULEVARD
KANSAS CITY, MO 64108-2613

EXAMINER	
CHOW, CHIH CHING	
ART UNIT	**PAPER NUMBER**
2191	

DATE MAILED: 11/03/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/990,527	SMITH, WALTER R.	
	Examiner	Art Unit	
	Chih-Ching Chow	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *07 September 2006*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-15* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-15* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *10 January 2005* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to amendment dated September 07, 2006.

2. Per Applicants' request, claims 1, 9-11 have been amended.

3. Claims 1-15 remain pending.

4. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 09/02/2005 has been entered.

Response to Amendment

5. Applicants' amendment for Claims 1, 9, 10, and 11 have been entered.

6. Applicants' amendment for Claims 1, 9, 10, and 11 have been fully considered. respectfully by the examiner but they are not persuasive.

7. The Examiner is maintaining the 35 USC § 103 Rejections. For the Applicants' convenience they are listed as following, with the amendments requested by the Applicants.

Response to Arguments

8. Applicants' arguments have been fully considered respectfully by the examiner but they are not persuasive.

9. Applicants' arguments are basically in the following points:

- "Clearly, programs instrumented using the technique discussed in O'brien are not ready for retail sale. Specifically, consumers generally do not own sophisticated probing hardware and certainly would not want to connect a probe when an application crashes. (REMARKS dated 6/2/06, pages 6-7).

Examiner's Response: In respond to the argument, the currently amended independent claims 1, 9, 10, and 11 recite: "prior to retail sale, removing each text string from the program; and releasing the program for retail sale" – it's obvious for the skilled people in the art to remove any instrumentation code from a program prior to retail sale, it's even recited in the current application, under the 'BACKGROUND OF THE INVENTION'.

Claim Rejections - 35 USC § 103

10. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

11. Claims 1, 3, 7-10, and 14 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,311,327 by Stephen Caine O'Brien et al. (hereinafter "O'Brien"), in view of current application (hereinafter "current").

CLAIM	O'Brien / Current
1. A method for obtaining information regarding events to be taking place within a software program to be used by a customer on a computing device, comprising: (a) including, for each of a number of selected events, an indicator within the software program that records the selected event, the indicator including a text string created by a software developer and descriptive of the selected event; (b) assigning and including a unique tag corresponding to each text string; (c) creating an index mapping each tag to the corresponding text string; and	In O'Brien, column 4, lines 9-12, "Data tags are always associated with a specific control tag, and they have a data field that provides information about an **event** identified by the **control tag** with which it is associated." (the selected *event identified by an indicator*). For items a, b, and c, see O'Brien, column 10, lines 49-60, "The parser 311 and the **tag** instrumenter 69 may be added as a new routine to the modified compiler 66 to **insert tag** statements at **appropriate points**.... The language-independent analyzer 321 may also be constructed as an information entry application program interface ("API"), according to an embodiment of the invention. An API is a library of **called**

(d) prior to retail sale, removing each text string from the program; and ~~prior to transferring the program to a customer~~

procedures used by an **application program**"(*in a software program*). Further, in O'Brien, column 11, lines 6-11, "the API 323 may provide a set of commands for retrieving information from the database 65 using a **tag value** as a **search key** ***(mapping index)***. Depending on the significance of the **tag** value, the API 323 may return a symbol name corresponding to the **tag**, a **text string**" (*a unique tag corresponding to each text string*). O'Brien teaches the concept of using a **unique tag** as an **index** key to a **corresponding text string** in a software program; the tag value can be inserted in **selected event** (*appropriate point, called procedure*). For item d, see O'Brien, column 9, lines 44-47, "The intermediate form of the AST 312 represents elementary processing of the source code 60 and, in some embodiments, may merely entail **removing programming comments from the source code** 60." And column 12, lines 3-5, "The C preprocessor 66a **removes information from the source code** 60 (*removing each test string prior to transferring the program to a customer*) such as comments that may have been added by the source code's programmer."; -- the information can include the text string from the program; and further, in column 22, lines 15-19, "A probe, such as the probe tip 12, represents but one mechanism for detecting **tags** during a program's execution. Other detection mechanisms include **writing tag values to a file** which for **subsequent analysis** and capturing **tag values** passing during an **external function call**." (see Fig. 1), this sentence implies that in O'Brien's art, the text string does not reside in the application program, the application

program only contains the **tag values**. Therefore, there is no need to remove text string before transferring the program to a customer. For item (e), O'Brien teaches all the aspect of claim 1, but he does not mention 'prior to retail sale' specifically, however, it's well known to the skilled people in the art to remove program instrumentation before the program to retail sale, even the current application recites this feature in the 'BACKGROUND OF THE INVENTION' section. See paragraph [0004], "Software developers make every effort to 'debug' the software programs before the programs are sold in the retail environment." And paragraph [0005], "**Prior to sending the program out into the retail environment**, however, the trace statements **are compiled out of the code**. Even if not compiled out of the code, it would be inefficient to use the trace log as a method of diagnosing and fixing bugs in the retail version of the software program. **Because the trace log is a human readable text file, it is very large**." It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement O'Brien's disclosure of the tagging of the application program by removing those instrumentation code prior to retail sale further taught by Current, for the purpose of economizing the size of the program (see paragraph [0005] of current application).

(e) releasing the program for retail sale.

3. The method of claim 1, wherein the indicator is a function call.

For the feature of claim 1 see claim 1 rejection. O'Brien teaches "capturing **tag values** passing during an **external function call**." (column 22, line 19). Therefore, the **indicator** (*tag value*) can be a **function call**.

7. The method of claim 1, further comprising including within selected indicators an identifier, the identifier identifying information unwanted by a software provider.	For the feature of claim 1 see claim 1 rejection. O'Brien's has disclosed the 'probe tip' (number 12 in Fig. 1) which is a **separate unit** from the computer, further, in column 7, lines 27-30, "After the probe chassis 20 has performed various **tabulation and data reduction functions** (*filtering out*) on the data from the probe tip 12, it outputs appropriate data to the host system 40 through the local area network cable 30". The probe contains an **identifier**, which will be used to **identify certain text string (identifying information)**, the text string is not used in the application program; this also means the identifier identifying information (matching text strings) are not required (*unwanted by a software provider*) for program execution, see 35 USC 112 rejection item 8 above.
8. The method of claim 7, further comprising: filtering out, prior to transmittal of the file to the repository, selected data indicated by the identifier as unwanted information.	For the feature of claim 7 see claim 7 rejection. Claim 7 rejection covers the 'filtering out' feature. In O'Brien, column 1, lines 48-51, "As another example, each tag statement may send **tag identifying data to a disk file** (*unwanted information; not delivered in software product*). As still another example, an array can be reserved in memory, with each array element corresponding to a tag inserted in a respective location in the source code." (***repository***).
9. A computer-readable medium having computer-executable instructions for performing a method for obtaining information regarding events to be taking place within a software program to be used by a customer on a computing device, comprising:	Same as claim 1 rejection. For the unique tag corresponding to each text string, see Obrien's column 1, lines 55-56, "an array can be reserved in memory, with **each array element corresponding to a tag** inserted in a respective location in the source code." – each tag is uniquely

searching for a text string within the software program created by a software developer and descriptive of a selected event; assigning and including a unique tag corresponding to each text string found; creating an index mapping each tag to the corresponding text string; ~~and~~ prior to retail sale, removing each text string from the program; and releasing the program for retail sale.	correspond to source code/text strings. For the 'prior to retail sale' feature, see claim 1 rejection.
10. A computer system having a processor, a memory, and an operating environment, the computer system operable to execute a method for obtaining information regarding events to be taking place within a software program to be used by a customer on a computing device, comprising: searching for a text string within the software program created by a software developer and descriptive of a selected event; assigning and including a unique tag corresponding to each text string found; creating an index mapping each tag to the corresponding text string; ~~and~~ prior to retail sale, removing each text string from the program; and releasing the program for retail sale.	Same as claim 1 rejection.
14. The method of claim 7, wherein the unwanted information is sensitive or personal information about the customer.	For the feature of claim 7 see claim 7 rejection. See O'Brien, column 12 lines 3-5, "The C preprocessor 66a removes information from the source code 60 such as comments that may have been added by the source code's programmer." – the information can be any information which is irrelated to the function implementation, such as sensitive or personal information about the customer.

12. Claims 2, 11-12, and 15 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,311,327 by Stephen Caine O'Brien et al. (hereinafter "O'Brien"), in view of current application (herein after "Current"), and further in view of US Patent No. 5, 608, 720 by Charles H. Biegel et al. (hereinafter "Biegel").

CLAIM	O'Brien / Current / Biegel
2. The method of claim 1, further comprising: (a) creating, on the computing device, a file of the recorded events including the unique tag for each event; (b) receiving, from the computing device, the file of the recorded events; (c) processing the file, by replacing into the file, the text string corresponding to each tag within the file; and (d) outputting a text string record of the events which took place within the software program, thereby providing a software provider a text record of the events taking place in the program to determine how the program may have failed.	For the feature of claim 1 see claim 1 rejection. For items a, b, and c see claim 1 rejection. For item d, O'Brien teaches providing text record when an error has occurred, in column 17, lines 22-26, "When an error is identified, a set of data and a control **tag** are written to indicate the **error**. The information present in the tags include an **error** identifier, the address of the block in **error** and its size (if any), the caller identifier(s) of the block's allocator and deallocator (if any), and the kind of allocator call begin attempted when the **error** was discovered."; but O'Brien does not mention the 'program may have failed' specifically. However, Biegel teaches this feature in an analogous art. In Biegel, column 28, lines 20-24, "An error identifier for an error is a **code** that is used to **decode the error** by offline tools in an output driver in the RDT (***coding and decoding***). This code is used to associate the error with a printable ASCII string.", column 48, lines 37-38, "If a **service operation fails**, the **failure** is translated into the most appropriate TL1 **error code**." It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement O'Brien's and Current disclosure of the tagging of the application program by the tagging the program failures further taught

by Biegel, for the purpose of aid in system debugging (see Biegel, column 29, lines 34-35).

11. A method for recording program information, by a software provider, about events to be taking place within a software program executing on a computer to be used by a customer, comprising:
(a) including, for each of a number of selected events, an indicator within the software program that records the selected event, the indicator including a text string created by a software developer and descriptive of the selected event;
(b) coding the text string with a unique tag corresponding to each text string;
(c) creating a decoding file mapping each unique tag to the corresponding text string; and
(d) prior to retail sale, removing each text string from the program; and ~~prior to transferring the program to a customer~~.
(e) releasing the program for retail sale.

O'Brien and Current teaches all aspects of claim 11 but does not mention the 'coding and decoding' (items b and c) specifically. However, Biegel teaches this feature, see claim 2 rejection (***coding and decoding***).

12. The method of claim 11, further comprising receiving, from the customer, a file of the recorded events, the file including the unique tag for each event;
decoding the file by mapping the coded tag with the corresponding text string; and
outputting a text string record of the events which took place within the software program,
thereby providing the software provider with a text record of the events taking place in the program to determine how the program may have failed.

For the feature of claim 11 see claim 11 rejection. For the rest of the feature of claim 12, see claim 2 rejection.

15. The method of claim 11, wherein removing each text string from the program

For the feature of claim 11 see claim 11 rejection. For the rest of the feature of

prior to transferring the program to a customer includes deleting each text string from the program but at least temporarily storing said each text string incident to said deleting.	claim 15, see claim 1 (d). Also see O'Brien's paragraph 0061, "The language-independent analyzer 321 determines a name, an identity, and appropriate reference numbers for inserted tags 62 and forwards this tagging information to the symbol database 65. The language-independent analyzer 321 receives programming context information from the C parser 69a and also stores this information in the symbol database 65 in an appropriate location for later reference." And paragraph 0087, "Calls to **an operator delete are followed by a call to the instrumented interface** (i.e., augmented-free), along with an appropriate memory management tag."

13. Claims 4-6, 13 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,311,327 by Stephen Caine O'Brien et al. (hereinafter "O'Brien"), in view of current application (hereinafter "Current"), further in view of US Patent No. 5, 608, 720 by Charles H. Biegel et al. (hereinafter "Biegel"), and further in view of U.S. Patent no. 5,245,615 by Albert R. Treu (hereinafter "Treu").

CLAIM	**O'Brien / Current / Biegel / Treu**
4. The method of claim 2, further comprising: as the program executes on the computing device, limiting the size of the file of the recorded events.	For the feature of claim 2 see claim 2 rejection. O'Brien and Biegel teach all aspects of claim 4 but does not mention the 'limiting the size of the file of the recorded events' specifically. However, Treu teaches this feature in an analogous art. In Treu, column 4, lines 66-67, "In a preferred embodiment, **error log** 88 has a **size** of 109 contiguous bytes." It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement O'Brien, Current, and Biegel's disclosure of the tagging of the application program

	and tagging the program failures by limiting the size of the error log taught by Treu, for the purpose of storing predetermined error log information at predetermined locations therein. (see Treu's Abstract, lines 2-3).
5. The method of claim 4, further comprising: in response to a failure of the program on the computing device, automatically transmitting the file to a repository accessible by the software provider.	For the feature of claim 4 see claim 4 rejection. For the 'transmitting the file to a repository' part, see claim 8 rejection.
6. The method of claim 5, wherein the failure is a crash of the program.	For the feature of claim 5 see claim 5 rejection. O'Brien and Treu teach the aspects of claim 6, except they don't mention 'crash' specifically. However, Biegel teaches the concept of crash as a failure of a program. In Biegel, column 27, lines 26-28, "4. **Crash** Log--captures a snapshot of the processor state when an irrecoverable error occurs on the processor". Therefore, it would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement O'Brien and Treu's disclosure of the tagging of the application program and tagging the program failures by tagging a crash condition taught by Biegel, for the purpose of capturing the state of the processor while a crash occurred (see Biegel, column 29, line 47).
13. The method of claim 2, further comprising: deleting the file when the program closes normally, without crashing.	For the feature of claim 2 see claim 2 rejection. O'Brien and Biegel teach all aspects of claim 13 but does not mention the 'deleting the file when the program does not have a crash' specifically. However, Treu teaches this feature in an analogous art. In Treu, column 8 last line to

column line 21, "**step 176 determines if an error or failure** (*crash*) occurred during the test. If not, step 178 then sees if any log entry has been saved by step 174, and if one has, step 180 informs the user that a temporary error may have occurred. ... If step 176 results in a positive determination, step 184 then compares the cause of failure with the log information saved in step 174. If they compare, step 188 **deletes the resource** (e.g. by deleting a block of memory from which the error arose) and informs the OS. Step 190 then deletes the log entry corresponding to such resource, builds an OS information log (error log status byte- bit 6) indicating the deletion and branches to step 182."
It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement O'Brien and Biegel's disclosure of the tagging of the application program and tagging the program failures by removing unnecessary error log taught by Treu, for the purpose of saving memory space for storing predetermined error log information at predetermined locations therein. (see Treu's Abstract, lines 2-3).

Conclusion

The following summarizes the status of the claims:

35 USC § 103 rejection: Claims 1-15

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Chih-Ching Chow whose telephone number is 571-272-3693. The examiner can normally be reached on 7:30am - 4:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached on 571-272-3708. The fax phone number for the

organization where this application or proceeding is assigned is 571-273-8300. Any inquiry of a general nature of relating to the status of this application should be directed to the **TC2100 Group receptionist: 571-272-2100.**

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Chih-Ching Chow

Examiner

Art Unit 2191

October 27, 2006

CC

WEI ZHEN
SUPERVISORY PATENT EXAMINER